Object-Oriented Design 2024L

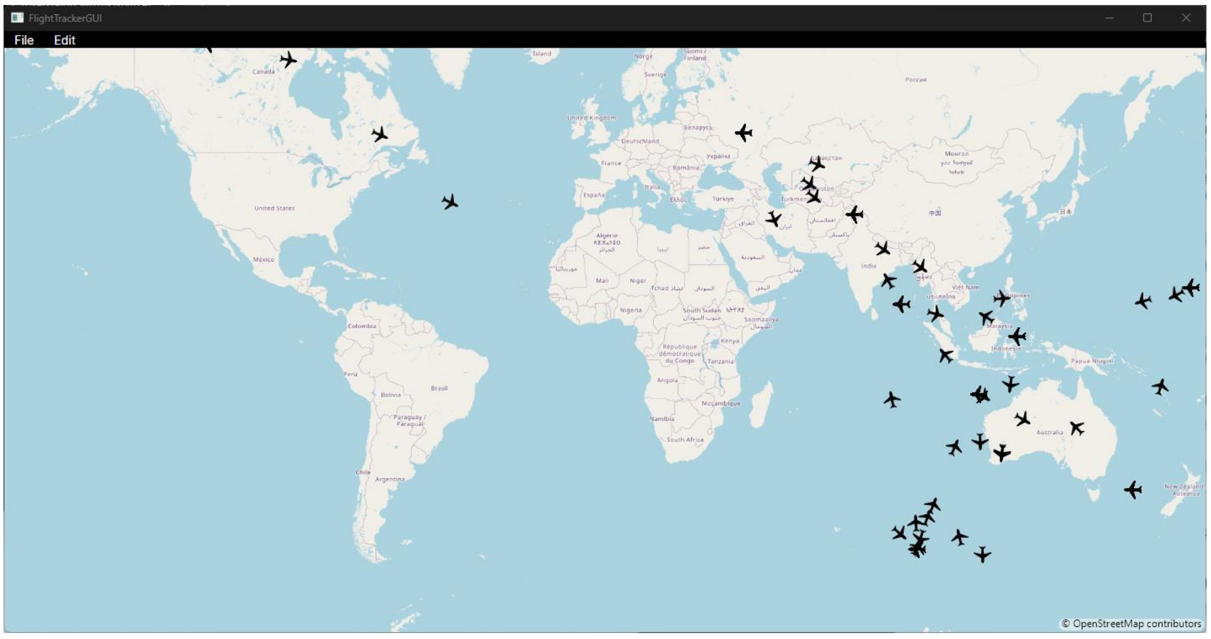
Stage 3 – Integration with GUI

Introduction

In this stage, the task is to add synchronization of data obtained from the server or text file with the graphical user interface (GUI).

The application with an interface has been prepared and can be imported into the project using the NuGet package. Knowledge of programming in a graphical environment is not necessary to display the application since all necessary methods for interacting with window controls have been extracted into separate functions.

Application appearance



The application window consists of a world map with icons representing the positions of airplanes at any given moment. All airplanes for which there is a flight in the stored database are displayed on the map. The positions of airplanes are constantly refreshed. The movement is minimal enough to recommend zooming in on the map to observe airplane movement.

The displayed map can be zoomed in using the mouse scroll.

Communication with the Application

P The NuGet package provides simplified wrappers for functions required to display the window and update airplane positions.

To achieve this, the following functions should be used:

- namespace FlightTrackerGUI:
  - → `void Runner.Run()` should be called to launch the application. This function launches the app and its loop.
  - → `void Runner.UpdateGUI(FlightsGUIData flightsGUIData)` should be called to send a new list of flights to the application window to refresh the displayed data.

The task involves writing a conversion between flight data from the `Flight` class and the expected `FlightGUI` data. It is also necessary to calculate the position of airplanes in the world (longitude and latitude) and rotation in the 2D space of the displayed map. The airplane's position should be calculated based on the location of the departure and destination airports – that is, the longitude and latitude of their positions. This value should be interpolated depending on the current time relative to the departure and arrival time of the airplane. The positions of airplanes should be updated every second. The direction of airplane flight can be calculated based on the current and previous locations, or the locations of origin and target airports.

To calculate the correct angle, transforming longitude and latitude into X, Y coordinates on the map might be helpful. For this purpose, you can use the function available in the imported NuGet package `MapsUI`:

```
(double x, double y) SphericalMercator.FromLonLat(double lon, double lan)
```

Class Structure of `FlightGUIData` and `FlightGUI`

Data accepted by the `UpdateGUI` function is in the form of an object of the class:

```
public class FlightsGUIData
{
      private List<FlightGUI> flightsData;
      virtual public int             GetFlightsCount();
      virtual public UInt64          GetID(int index);
      virtual public WorldPosition   GetPosition(int index);
      virtual public double          GetRotation(int index);
}
```

The objects in the internal list `flightsData` have the form of objects of the class:

```
public class FlightGUI

{

      public UInt64           ID { get; init; }

      public WorldPosition    WorldPosition { get; init; }

      public double           MapCoordRotation { get; init; }

}
```

`ID` is a unique flight identifier. `WorldPosition` is a structure storing latitude and longitude. `MapCoordRotation` is the angle between the flight direction vector in map coordinates and the vector (0,1), given in radians. Simplifying, when the angle is 0, the airplane is directed upwards on the map, for an angle of PI/2 - to the right, and so on.

Deadline

2 weeks

All source files must be uploaded to the Git repository by 26.03.2024 23:59.

The project should be presented to the instructor during the class on 27.03.2024.